PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : **H04L 12/56** — **A2**

(11) International Publication Number: **WO 99/13619**

(43) International Publication Date: 18 March 1999 (18.03.99)

(21) International Application Number: PCT/SE98/01584

(22) International Filing Date: 7 September 1998 (07.09.98)

(30) Priority Data:
9703292–4 9 September 1997 (09.09.97) SE

(71) Applicant *(for all designated States except US)*: SICS SWEDISH INSTITUTE OF COMPUTER SCIENCE [SE/SE]; P.O. Box 1263, S–164 29 Kista (SE).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: SJÖDIN, Peter [SE/SE]; Nyodlingsvägen 14B, S–191 40 Sollentuna (SE). MOESTEDT, Andreas [SE/SE]; Fatburskvarnsgata 2, S–118 64 Stockholm (SE).

(74) Agent: ASKERBERG, Fredrik; L.A. Groth & Co. KB, P.O. Box 6107, S–102 32 Stockholm (SE).

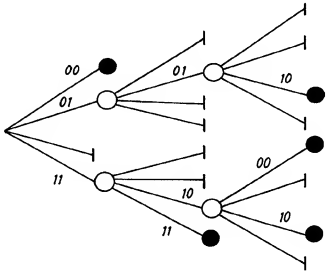
(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, HR, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published
Without international search report and to be republished upon receipt of that report.

(54) Title: A LOOKUP DEVICE AND A METHOD FOR CLASSIFICATION AND FORWARDING OF PACKETS

(57) Abstract

The present invention relates to a lookup device (30) and a method for classification and forwarding of packets. The lookup device (30) comprises i stages (32_1, 32_2, 32_3, 32_4) wherein each stage (32_1, 32_2, 32_3, 32_4) represents a predetermined prefix length, and a prefix represents a group of addresses. The lookup device (30) also comprises a routing memory means (34) connected to said i:th stage (32_4), wherein each stage (32_1, 32_2, 32_3, 32_4) comprises a memory means (36_1, 36_2, 36_3, 36_4) for storing a table of entries, wherein each entry comprises a pointer field which either comprises a pointer to the memory means in the next stage, or a pointer to the routing memory means (34) storing the forwarding information.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

A LOOKUP DEVICE AND A METHOD FOR CLASSIFICATION AND FORWARDING OF PACKETS

Technical field of the invention

The present invention relates to a lookup device and a method for classification and forwarding of packets.

Description of related art

The growth of the Internet has led to a situation where bandwidth is becoming a scarce resource. One reason for this is that the IP routers - the packet switches in the Internet - are not powerful enough to handle the traffic that aggregates at the switching points. The current trend for dealing with this problem is to relieve routers from some of the burden of switching traffic, and instead use switches of different kinds, such as FDDI switches, ATM switches, and Ethernet switches. This turns out to be a more cost effective solution, since the price for switching capacity is much lower than the price for routing capacity.

One of the main limiting factors for performance in an IP router, compared to a switch is often claimed to be the processing of incoming packets. When an IP packet arrives at an input port of a router, the packet needs to be examined and classified, and based on the classification the packet is forwarded to an output port. The packet classification operation consists of analysing information in the packet header (at least the destination address needs to be examined), and performing a lookup operation to obtain the information required to forward the packet to its next hop. In principle, the same kind of classification needs to be performed by a switch, but the operation is generally thought to be more complicated for an IP packet than for an ATM cell or an Ethernet frame. The lookup operation consists of searching a database for an entry matching the packet. The efficiency of a lookup operation depends on the data structure used for the database. Data structures such as simple tables and linked lists are easy to implement, but have the drawback that they are difficult to search efficiently.

An interesting compromise between a simple table and a linked list is a trie. In this approach, the address is partitioned into a number of sections, and each section is used to address a different level of a search tree. At each level, a number of small tables are used to store pointers to the appropriate tables on the next level. There is still wasted space within each of the tables, but only a partial tree need be created, covering the portions of the address space that are in use at any one time. The trie is most efficient when the utilized addresses are clustered in the address space.

The article "VLSI Implementation of Routing Tables: Tries and CAMs", by T. Pei, C. Zukowski, Proceedings of INFOCOM '91, 991, pp. 0515-0524, discloses the use of tries in routing tables, for address lookup with fixed lengths.

Binary tries have also been used for variable length addresses, for example the commonly used Patricia trie, which is a refinement of a binary trie. But since IP addresses are 32 bits long, a fast hardware implementation of Patricia tries is expensive and complex.

There have been several suggestions to implement routing table lookups by using Content Addressable Memory (CAM) to store the routing table. However, CAM technology does not currently provide large memory at a sufficiently low cost to make this a practical solution.

Summary of the invention

The object of the present invention is to solve the above mentioned problems and to provide a lookup device for classification and forwarding of packets, wherein the input to the lookup device is a destination address of an incoming packet and the destination address comprises n bits, wherein n is an integer. This object is achieved by providing the lookup device defined in the introductory part of Claim 1 with the advantageous features of the characterizing part of said Claim.

The lookup device according to the present invention comprises i stages, wherein each stage represents a predetermined prefix length and a prefix represents a group of addresses, wherein the first stage represents the shortest prefix length and the i:th stage represents the longest prefix length with n bits. The lookup device also comprises a routing memory means connected to said i:th stage, wherein each stage comprises a memory means for storing a table of entries, wherein each entry comprises a pointer field which either comprises a pointer to the memory means in the next stage, or a pointer to the routing memory means storing the forwarding information, wherein a first number of bits corresponding to the shortest prefix length of the destination address are searched for in the first memory means, and if the entry in the first memory means wherein a match is found comprises a pointer to the second memory means, then a second number of bits corresponding to the next shortest prefix length of the destination address are searched for in the second memory means, and so on until a longest matching prefix has been found, wherein the pointer to the routing memory means gives the forwarding information for forwarding of said incoming packet. The main advantage with this design is that it is capable of performing one lookup per memory cycle. Furthermore the design is simple and fast, and it is therefore suitable for a high-end IP router. This solution also works with identifiers with different lengths.

Advantageously, the first stage comprises only said first memory means, and all the other stages comprise each a logic means, wherein the (q-1):th logic means is connected to both the (q+1):th memory means and the q:th memory means, wherein q is an integer and $1 \leq q \leq i-1$.

Preferably, each entry in the first (i-1) memory means also comprises a part bit and a valid bit, and each entry in the i:th memory means also comprises a valid bit, wherein a set valid bit in an entry represents a valid prefix and the pointer field for that entry comprises a

pointer to the routing memory means, and a set part bit in an entry in the q:th memory means means that the prefix matches an entry in the routing memory means, but is shorter than that entry, and the pointer field for that entry in the q:th memory means comprises a pointer to the (q+1):th memory means.

Advantageously, each of the first (i-1) memory means has a pointer output, a valid bit output and a part bit output, and the i:th memory means has a pointer output and a valid bit output, wherein the pointer output of the q:th memory means is connected to the (q+1):th memory means, and to the logic means of the (q+1):th stage, and wherein the valid bit output and the part bit output of the q:th memory means are connected to the logic means of the (q+1):th stage.

Preferably, the logic means in the i:th stage comprises a multiplexer, an OR-gate, and an AND-gate, and each logic means in all the other stages but the first stage comprises a multiplexer, an OR-gate, a first AND-gate, and a second AND-gate, wherein the inputs of the multiplexer in the second stage are connected to the pointer outputs of the first and second memory means, and the inputs of the first AND-gate in the second stage are connected to the valid bit output of the second memory means and to the part bit output of the first memory means, and the inputs of the second AND-gate in the second stage are connected to the part bit outputs of the fist and second memory means, and the inputs of the OR-gate in the second stage are connected to an output from the first AND-gate in the second stage and to the valid bit output of the first memory means, wherein the inputs to the multiplexer in the p:th stage, wherein p is an integer and $3 \leq p \leq i-1$, are connected to the pointer output of the p:th memory means and to an output of the multiplexer of the (p-1):th stage, and the inputs of the first AND-gate in the p:th stage are connected to the valid bit output of the p:th memory means and to an output of the second AND-gave in the (p-1):th stage, and the inputs of the second

AND-gate in the p:th stage are connected to the part bit output of the p:th memory means and to the output of the second AND-gate in the (p-1):th stage, and the inputs of the OR-gate in the p:th stage are connected to the output of the first AND-gate in the (p-1):th stage and to an output of the OR-gate in the (p-1):th stage, and wherein the inputs of the multiplexer in the i:th stage are connected to the pointer output of the i:th memory means and to an output of the multiplexer of the (i-1):th stage, and the inputs of the AND-gate in the i:th stage are connected to the valid bit output of the i:th memory means and to the output of the second AND-gate in the (i-1):th stage, and the inputs of the OR-gate in the i:th stage are connected to the output of the AND-gate in the i:th stage and to the output of the OR-gate in the (i-1):th stage.

Advantageously, a match is found when an entry in one stage comprises a set valid bit output and all the matching entries of shorter prefix lengths are set part bit outputs.

Preferably, each stage also comprises a decompression logic means, wherein the i:th decompression logic means is connected to the multiplexer in the i:th stage, and the q:th decompression logic means is connected to both the q:th memory means and to the (q+1):th memory means. Hereby is achieved a memory saving optimization.

Advantageously, each entry in each memory means also comprises a mask field, an address tag field, and a compress flag bit, wherein a set compress flag bit indicates that compression is used, and in that each of the memory means also has a mask field output, an address tag field output, and a compress flag bit output, wherein the mask field output, the address tag field output, the compress flag bit output, and the pointer output of the r:th memory means is connected to the r:th decompression logic means, wherein r is an integer and $1 \leq r \leq i$, and wherein each decompression logic means outputs a new pointer, wherein the new pointer output from the q:th decompression logic means is connected to the (q+1):th

memory means and the new pointer output from the i:th decompression logic means is connected to the multiplexer of the i:th stage.

Preferably, each decompression logic means comprises an AND-gate, a comparator, a first multiplexer, and a second multiplexer, wherein the inputs of the AND-gate in the r:th decompression logic means is connected to the mask field output of the r:th memory means and to the r:th prefix length of the destination address, and the inputs of the comparator in the r:th decompression logic means is connected to the address tag field output of the r:th memory means and to an output from the AND-gate in the r:th decompression logic means, and the inputs of the first multiplexer in the r:th decompression logic means is connected to the pointer output of the r:th memory means and to a logically 0-signal, and the inputs of the second multiplexer in the r:th decompression logic means is connected to the output from the AND-gate in the r:th decompression logic means, and to an output from the comparator in the r:th decompression logic means, and in that the new pointer output from the r:th decompression logic means either is the output from the first multiplexer or the output from the second multiplexer in the r:th decompression logic means depending on if the compress flag bit is sets or not.

Another object of the invention is to provide a method for classification and forwarding of packets, wherein a destination address of an incoming packet comprises bits, wherein n is an integer. This object is achieved by providing the method defined in the introductory part of Claim 10 with the advantageous features of the characterizing part of said Claim.

The method according to the present invention comprises i stages, wherein each stage represents a predetermined prefix length and a prefix represents a group of addresses, wherein the first stage represents the shortest prefix length and the i:th stage represents the longest prefix length with n bits, wherein each stage

comprises a table of entries, wherein each entry comprises a pointer field which either comprises a pointer to the table in the next stage, or a pointer to a routing table storing the forwarding information, wherein the method comprises the steps:

- to search for a first number of bits corrresponding to the shortest prefix length of the destination address in the first table;
- if the entry in the first table wherein a match is found comprises a pointer to the second table, to search for a second number of bits corresponding to the next shortest prefix length of the destination address in the second table;
- and so on until a longest matching prefix has been found, wherein the pointer to the routing table gives the forwarding information for forwarding of said incoming packet. The main advantage with this method is that it is capable of performing one lookup per table cycle. Furthermore, the method is simple and fast, and it is therefore suitable for a high-end IP router.

Advantageously, each entry in the first (i-1) tables also comprises a part bit and a valid bit, and each entry in the i:th table also comprises a valid bit, wherein a set valid bit in an entry represents a valid prefix, and the pointer field for that entry comprises a pointer to the routing table, and a set part bit in an entry in the q:th table means that the prefix matches an entry in the routing table, but is shorter than that entry, and the pointer field for the entry in the q:th table comprises a pointer to the (q+1):th table, wherein 1 is an integer and $1 \leq q \leq i-1$.

Preferably, an entry in the first (i-1) tables is not allowed to both have a set valid bit and a set part bit.

Advantageously , a match is found when an entry in one stage comprises a set valid bit and all the matching entries of shorter prefix lengths are set part bits.

Preferably, each entry in each table also comprises a mask field, an address tag field, and a compress flag bit,

wherein a set compress flag bit indicates that compression is used, wherein the method for the r:th table, wherein r is an integer and $1 \leq r \leq i$, also comprises the following steps:

- to AND-process the mask field and the destination address bits corresponding to the r:th prefix length, giving masked destination address bits as output;
- if the compress flag bit is not set, to output a new pointer which is input to the g:th table, and which is input to the routing table for the i:th table, wherein the new pointer is formed by using the pointer from the (q-1):th table unchanged, with the destination address bits corresponding to the r:th prefix length as low order bits; or
- if the compress flag bit is set, to output a new pointer which is input to the g:th table, and which is input to the routing table for the i:th table, wherein the new pointer is formed with the high order bits set to 0, and the low order bits are formed using the bits of the igh order part of the pointer from the (g-1):th table, and the least significant bit of the new pointer comes from the comparing step.

Embodiments of the invention will now be described with a reference to the accompanying drawings, in which:

Brief Description of the Drawings

Figure 1 shows a schematic diagram of the fields in an IP packet header;

Figure 2 shows a schematic diagram of a prefix tree with 6-bit addresses using three prefix lengths according to the principle of the present invention;

Figure 3 shows a block diagram of a lookup device according to the present invention;

Figure 4 shows a block diagram of a decompression logic means forming part of the lookup device according to the present invention; and

Figure 5 is a flow chart of the method according to the present invention.

Detailed Description of Embodiments

In figure 1 there is disclosed a schematic diagram of the fields in an IP packet header. The IP packet header comprises 12 different fields. As is disclosed in figure 1 these fields are: Version, IP Header Length, Type of Service, Total Length, Identification, Flags, Fragment Offset, Time to Live, Protocol, Header Checksum, Source Address, and Destination Address. It can also contain an Options field.

There are in principle two different types of IP packet classification: IP address lookup, which is used for forwarding of unicast packets based on their destination address, and identifier lookup, which is intended to be used for, for example, forwarding of multicast packets and flows of packets.

The present invention is based on the IP address lookup.

A routing table entry for IP addresses has two fields; a prefix and a next hop address. A prefix represents a group of addresses, and is given as an IP address prefix and a prefix length. For example, the IP prefix 193.10.66/234 represents all IP addresses whose 24 first bits are equal to 193.10.66. So a routing table entry for a given prefix applies to all addresses which are in the group covered by the prefix.

The principle for an address lookup is based on the longest matching prefix; with the destination address of an incoming IP packet as key, the routing table is searched for entries with matching prefixes. If there are more than one matching entry, the one with the longest prefix is chosen. By using this principle, it is always the most specific routing table entry that is picked. For example, consider a routing table with the following three entries:

Prefix	Next hop
0/0	193.10.66.1
193.10.64/26	193.10.66.27
193.10.66/28	193.10.66.138

The address 193.10.66.50 would match all three entries, but it is the last entry (193.10.66/28) that is picked, since it has the longest prefix (i.e., it is the most specific entry).

In figure 2 there is disclosed a schematic diagram of a prefix tree with 6-bit addresses using three prefix lengths according to the principle of the present invention. The address space can be thought of as a tree, where the nodes represent prefixes. Each level in the tree represents a specific prefix length. Prefixes with other lengths than the ones used in the tree, have to be extended to several longer prefixes. For example, for the tree in figure 2 the prefix "010" of length 3 would have to be expanded into "0100" and "0101" of length 4.

In the tree, each node has one of two attributes, the first indicates if the node represents a valid prefix, corresponding to an entry in the routing table, the second indicates if it is part of a valid prefix, being a part means that the prefix matches an entry in the routing table, but is shorter than that entry (a prefix of a prefix), a node is not allowed to be both valid and part. If it is, the valid route has to be extended to the next length, marking all nodes of this length as valid. If none of the attributes are set, the node is said to be invalid.

To find a matching route, the tree is searched from the shortest prefix until the first valid or invalid note is encountered. In this way the longest match is guaranteed to be found.

The following is the procedure to find the address "010110" in a routing table represented by the tree in figure 2. The three prefix lengths are 2, 4, and 6. First the shortest prefix length "01" is looked up. The matching entry is a part, resulting in a second lookup, "0101". This entry is also a part. A last lookup "010110" is then performed, resulting in a valid entry. This entry points into the routing table where the forwarding information is stored.

When fewer prefix levels are used in the data structure, more memory is needed to store it. This is due to the bit extension up to a valid prefix length, and the need for all entries in a prefix group to be present. If, for example, the step between two lengths is 8 bits, every node that is the part attribute set will have 256 child nodes in its prefix group. But with fewer levels, a prefix is found with fewer memory accesses, making the lookup faster. The ideal choice of levels and the distance between, depends on what performance is needed, and how much memory can be used.

In figure 3 there is disclosed a block diagram of a lookup device according to the present invention. The lookup device 30 is based on IP address lookup, which is used for forwarding of unicast packet based on their destination address. The lookup device 30 comprises i stages 32_1, 322, ...32_i. In this figure i is 4. The first stage 32_1 represents the shortest prefix length and the 4:th stage 32_4 represents the longest prefix length with n bits. The lookup device 30 also comprises a routing memory means connected to said 4:th stage 32_4. Each stage 32_1, 32_2, 32_3, 32_4 comprises a memory means 36_1, 36_2, 36_3, 36_4 for storing a table of entries, wherein each entry comprises a pointer field which either comprises a pointer to the memory means in the next stage, or a pointer to the routing memory means 34 which stores the forwarding information. The destination address is input to the lookup device 30 via a line 38. A first number of bits corresponding to the shortest prefix length of the destination address are searched for in the first memory means 36_1, and if the entry in the first memory means 36_1 wherein a match is found comprises a pointer to the second memory means 36_2, then a second number of bits corresponding to the next shortest prefix length of the destination address are searched for in the second memory means 36_2, and so on until a longest matching prefix has been found, wherein the pointer to the routing memory

means 34 gives the forwarding information for forwarding of said incoming packet. All stages but the first 32_2, 32_3, 32_4 each comprises a logic means 40_2, 40_3, 40_4, wherein the i:th logic means is connected to both the i:th memory means and the (i-1):th memory means. Each entry in the 4:th memory means 36_4 also comprises a valid bit, and each entry in the 3 first memory means 36_1, 36_2, 36_3 also comprises a part bit and a valid bit. A set valid bit in an entry represents a valid prefix, and the pointer field for that entry comprises a pointer to the routing memory means 34. A set part bit in an entry in the q:th memory means 36_1, 36_2, 36_3 means that the prefix matches an entry in the routing memory means 34, but is shorter than that entry, and the pointer field for the entry in the q:th memory means 36_1, 36_2, 36_3 comprises a pointer to the (q+1):th memory means 36_2, 36_3, 36_4 wherein q is an integer and $1 \leq q \leq i-1=3$. Each of the three first memory means 36_1, 36_2, 36_3 has a pointer output, a valid bit output and a part bit output. The 4:th memory means 36_4 has only a pointer output and a valid bit output. The 4:th logic means 40_4 comprises a multiplexer 42_4, an OR-gate 44_4 and an AND-gate 46_4. Each logic means in all the other stages but the first stage (40_2, 40_3) comprises a multiplexer 42_2; 42_3, and an OR-gate 44_2; 44_3, a first AND-gate 46_2; 46_3 and a second AND-gate 48_2; 48_3. The inputs of the multiplexer 42_2 are connected to the pointer outputs of the first and second memory means 36_1, 36_2. The inputs of the first AND-gate 46_2 are connected to the valid bit output of the second memory means 36_2 and to the part bit output of the first memory means 36_1. The inputs of the second AND-gate 48_2 are connected to the part bit output of the first and second memory means 36_1, 36_2. The inputs of the OR-gate 44_2 are connected to an output of the first AND-gate 46_2 and to the valid bit output of the first memory means 36_1. The inputs to the multiplexer in the p:th stage, wherein p is an integer and $3 \leq p \leq i-1$, are connected to the pointer output of the p:th memory means and to an output of the multiplexer of the (p-1):th stage.

The inputs of the first AND-gate in the p:th stage are connected to the valid bit output of the p:th memory means and to an output of the second AND-gate in the (p-1):th stage. The inputs of the second AND-gate in the p:th stage are connected to the partbit output of the p:th memory means and to the output of the second AND-gate in the (p-1):th stage. The inputs of the OR-gate in the p:th stage are connected to the output of the first AND-gate in the (p-1):th stage and to an output of the OR-gate in the (p-1):th stage. The inputs of the multiplexer 42_4 are connected to the pointer output of the memory means 36_4 and to an output of the multiplexer 42_3. The inputs of the AND-gate 46_4 are connected to the valid bit output of the memory means 36_4 and to the output of the second AND-gate 48_3. The inputs of the OR-gate 44_4 are connected to the output of the AND-gate 46_4 and to the output of the OR-gate 44_3. The routing memory means 34 are connected to the outputs from the multiplexer 42_4 and the OR-gate 44_4.

The part bit and the valid bit are used to determine when the longest match is found. In short, a match is found if an entry in one stage is valid and all the matching entries of shorter lengths are parts. The pointer and the part and valid bits are sent from one stage to the next until all lengths have been examined. When a match is found, the resulting router table pointer flows through the following stages, without being changed. The multiplexer selects the pointer of its stage if the valid bit of that stage is set, and the part bit of all previous stages are set. If this is not true the multiplexer selects the output from the previous stage, i.e. sends the results on from an earlier hit.

In figure 4 there is disclosed a block diagram of a decompression logic means included in the lookup device according to the present invention. There is only disclosed one decompression logic means 50_1, but each stage in the lookup device 30 (see figure 3) comprises a decompression logic means. In the lookup device 30 according to figure 3, there should be 4 decompression

logic means 50_1, 50_2, 50_3, 50_4. Each decompression logic means 50_1 comprises an AND-gate 52_1, a comparator 54_1, a first multiplexer 56_1 and a second multiplexer 58_1. Each entry in each memory means 36_1, 36_2, 36_3, 36_4 (see figure 3) also comprises a mask field, and address tag field, and a compress flag bit, wherein a set compress flag bit indicates that compression is used. Each of the memory means 36_1, 36_2, 36_3, 36_4 also has a mask field output, an address tag field output, and a compress flag bit output. In figure 4 there is disclosed the decompression logic means 50_1 for the first stage. The decompression logic means for the other stages are constructed in the same way and are not disclosed. The inputs to the AND-gate 52_1 are the mask field output and the first prefix length of the destination address. The inputs to the comparator 54_1 are the address tag field output and an output from the AND-gate 52_1. The inputs to the first multiplexer 56 are the pointer output and a logically 0-signal. The inputs to the second multiplexer are the output from the AND-gate 52_1 and an output from the comparator 54_1. The decompression logic means 50_1 outputs a new pointer, which is input to the second memory means 36_2 (see figure 3).

The background to the design according to figure 4 is that a common case in sparse routing tables is that a prefix group only contains two different kinds of entries. The first kind is one or several consecutive valid entries, all identical. The second kind consists of all other entries in the group and are either invalid or valid. If they are valid, it is due to the extension of a shorter matching route, that the first kind of entries overlap. To optimize the memory requirements for these groups, we introduce a special way of representing them. To each part entry a tag is added which identifies the valid entry of the following length, and also a length indicator or a mask to allow groups of valid. In this way the table can be reduced to only two entries. One entry for addresses that match the tag, and one entry for the other addresses. Also needed is one bit to flag that the

compression is to be done. Bits in the entries are saved by placing all the compressed tables in the low address range of the memory. By doing this, the high order address bits can instead be used as low order bits when addressing among the compressed tables.

In figure 5 there is disclosed a flow chart of the method according to the present invention. The method begins at block 60. The method comprises i stages, wherein each stage represents a predetermined prefix length and a prefix represents a group of addresses, wherein the first stage represents the shortest prefix length and the i:th stage represents the longest prefix length with n bits, wherein each stage comprises a table of entries, wherein each entry comprises a pointer field which either comprises a pointer to the table in the next stage, or a pointer to a routing table storing the forwarding information. Thereafter, at block 62, the method continues to search for a first number of bits corresponding to the shortest prefix length of the destination address in the first table. Thereafter, at block 64, the question is asked whether the entry in the first table wherein a match is found comprises a pointer to the second table. If the answer is affirmative the method continues with block 66, wherein a search is performed for a second number of bits corresponding to the next shortest prefix length of the destination address in the second table. Thereafter, at block 68, the question is asked whether the entry in the second table wherein a match is found comprises a pointer to the third table. If the answer is affirmative the method continues until a longest matching prefix has been found, wherein the pointer to the routing table, at block 70, gives the forwarding information for forwarding of said incoming packet. Then the method is completed at block 72. If the answer at block 64 or 68 is negative the method continues with block 70.

The invention is not limited to the embodiment described in the foregoing. It will be obvious that many different modifications are possible within the scope of the following Claims.

Claims

1. A lookup device (30) for classification and forwarding of packets, wherein the input to the lookup device (30) is a destination address of an incoming packet and the destination address comprises n bits, wherein n is an integer **characterized in** that the lookup device comprises i stages (32_1, 32_2, 32_3, 32_4), wherein each stage (32_1, 32_2, 32_3, 32_4) represents a predetermined prefix length and a prefix represents a group of addresses, wherein the first stage (32_1) represents the shortest prefix length and the i:th stage (32_4) represents the longest prefix length with n bits, and in that the lookup device (30) also comprises a routing memory means (34) connected to said i:th stage (32_4), wherein each stage (32_1, 32_2, 32_3, 32_4) comprises a memory means (36_1, 36_2, 36_3, 36_4) for storing a table of entries, wherein each entry comprises a pointer field which either comprises a pointer to the memory means in the next stage, or a pointer to the routing memory means (34) storing the forwarding information, wherein a first number of bits corresponding to the shortest prefix length of the destination address are searched for in the first memory means (36_1), and if the entry in the first memory means (36_1) wherein a match is found comprises a pointer to the second memory means (36_2), then a second number of bits corresponding to the next shortest prefix length of the destination address are searched for in the second memory means (36_2) and so on until a longest matching prefix has been found, wherein the pointer to the routing memory means (34) gives the forwarding information for forwarding of said incoming packet.

2. A lookup device (30) according to Claim 1, **characterized in** that the first stage (32_1) only comprises said first memory means (36_1), and in that all the other stages (32_2, 32_3, 32_4) each comprises a logic means (40_2, 40_3, 40_4), wherein the (q+1):th logic means (40_2, 40_3, 40_4)

is connected to both the (q+1):th memory means (36_2, 36_3, 36_4) and the 1:th memory means (36_1, 36_2, 36_3), wherein q is an integer and $1 \leq q \leq i-1$.

3. A lookup device (30) according to Claim 2, **characterized in** that each entry in the first (i-1) memory means (36_1, 36_2, 36_3) also comprises a part bit and a valid bit, and each entry in the i:th memory means (36_4) also comprises a valid bit, wherein a set valid bit in an entry represents a valid prefix, and the pointer field for that entry comprises a pointer to the routing memory means (34), and a set part bit in an entry in the q:th memory means means that the prefix matches an entry in the routing memory means (34), but is shorter than that entry, and the pointer field for the entry in q:th memory means (36_1, 36_2, 36_3) comprises a pointer to the (q+1):th memory means (36_2; 36_3; 36_4).

4. A lookup device (30) according to Claim 3, **characterized in** that each of the first (i-1) memory means (36_1, 36_2, 36_3) has a pointer output, a valid bit output and a part bit output, and in that the i:th memory means (36_4) has a pointer output and a valid bit output, wherein the pointer output of the q:th memory means (36_1; 36_2; 36_3) is connected to the (q+1):th memory means (36_2; 36_3; 36_4), and to the logic means of the (q+1):th stage (32_2, 32_3, 32_4), and wherein the valid bit output and the part bit output of the q:th memory means (36_1, 36_2, 36_3) are connected to the logic means of the (q+1):th stage (32_2, 32_3, 32_4).

5. A lookup device (30) according to Claim 4, **characterized in** that the logic means (40_4) in the i:th stage (32_4) comprises a multiplexer (42_4) , an OR-gate (44_4), and an AND-gate (46_4), and in that each logic means (40_2; 40_3) in all the other stages but the first stage comprises a multiplexer (42_2; 42_3), an OR-gate (44_2; 44_3), a first AND-gate (46_2; 46_3), and a second AND-gate (48_2, 48_3), wherein the inputs of the multiplexer (42_2) in the

second stage are connected to the pointer outputs of the first and second memory means (36_1, 36_2), and the inputs of the first AND-gate (46_2) in the second stage are connected to the valid bit output of the second memory means (36_2) and to the part bit output of the first memory means (36_1), and the inputs of the second AND-gate (48_2) in the second stage are connected to the part bit outputs of the first and second memory means (36_1, 36_2), and the inputs of the OR-gate (44_2) in the second stage are connected to an output from the first AND-gate (46_2) in the second stage and to the valid bit output of the first memory means (36_1), wherein the inputs to the multiplexer (42_3) in the p:th stage, wherein p is an integer and $3 \leq p \leq i-1$, are connected to the pointer output of the p:th memory means (36_3) and to an output of the multiplexer (42_2) of the (p-1):th stage, and the inputs of the first AND-gate (46_3) in the p:th stage are connected to the valid bit output of the p:th memory means (36_3) and to an output of the second AND-gate (48_2) in the (p-1):th stage, and the inputs of the second AND-gate (48_3) in the p:th stage are connected to the part bit output of the p:th memory means (36_3) and to the output of the second AND-gate (48_2) in the (p-1):th stage, and the inputs of the OR-gate (44_3) in the p:th stage are connected to the output of the first AND-gate (46_3) in the p:th stage and to an output of the OR-gate (44_2) in the (p-1):th stage, and wherein the inputs of the multiplexer (42_4) in the i:th stage are connected to the pointer output of the i:th memory means (36_4) and to an output of the multiplexer (42_3) of the (i-1):th stage, and the inputs of the AND-gate (46_4) in the i:th stage are connected to the valid bit output of the i:th memory means (36_4) and to the output of the second AND-gate (48_3) in the (i-1):th stage, and the inputs of the OR-gate (44_4) in the i:th stage are connected to the output of the AND-gate (46_4) in the i:th stage and to the output of the OR-gate (44_3) in the (i-1):th stage.

6. A lookup device (30) according to Claim 5, **characterized in** that a match is found when an entry in one stage comprises a set valid bit output and all the matching entries of shorter prefix lengths are set part bit outputs.

7. A lookup device (30) according to any one of Claims 1-6, **characterized in** that each stage (32_1, 32_2, 32_3, 32_4) also comprises a decompression logic means (50_1, 50_2, 50_3, 50_4), wherein the i:th decompression logic means (50_4) is connected to the multiplexer (42_4) in the i:th stage and the q:th decompression logic means (50_1; 50_2; 50_3) is connected to both the q:th memory means (36_1; 36_2; 36_3) and to the (q+1):th memory means (36_2; 36_3; 36_4).

8. A lookup device (30) according to Claim 7, **characterized in** that each entry in each memory means (36_1, 36_2, 36_3, 36_4) also comprises a mask field, an address tag field, and a compress flag bit, wherein a set compress flag bit indicates that compression is used, and in the each of the memory means (36_1, 36_2, 36_3, 36_4) also has a mask field output, an address tag field output, and a compress flag bit output, wherein the mask field output, the address tag field output, the compress flag bit output, and the pointer output of the r:th memory means (36_1; 36_2; 36_3; 36_4) is connected to the r:th decompression logic means (50_1; 50_2; 50_3; 50_4) wherein r is an integer and $1 \leq r \leq i$, and wherein each decompression logic means (50_1, 50_2, 50_3, 50_4) outputs a new pointer, wherein the new pointer output from the q:th decompression logic means (50_1; 50_2; 50_3) is connected to the (q+1):th memory means (36_2; 36_3; 36_4) and the new pointer output from the i:th decompression logic means (50_4) is connected to the multiplexer (42_4) in the i:th stage.

9. A lookup device (30) according to Claim 8, **characterized in** that each decompression logic means (50_1; 50_2; 50_3; 50_4) comprises an AND-gate (52_1; 52_2; 52_3; 52_4), a comparator (54_1; 54_2; 54_3; 54_4), a first multiplexer (56_1;

56_2; 56_3; 56_4) and a second multiplexer (58_1; 58_2; 58_3; 58_4), wherein the inputs of the AND-gate (52_1; 52_2; 52_3; 52_4) in the r:th decompression logic means (50_1; 50_2; 50_3; 50_4), is connected to the mask field output of the r:th memory means (36_1; 36_2; 36_3; 36_4) and to the r:th prefix length of the destination address, and the inputs of the comparator (54_1; 54_2; 54_3; 54_4) in the r:th decompression logic means (50_1; 50_2; 50_3; 50_4) is connected to the address tag field out put of the r:th memory means (36_1; 36_2; 36_3; 36_4) and to an output from the AND-gate (52_1; 52_2; 52_3; 52_4) in the r:th decompression logic means (50_1; 50_2; 50_3; 50_4), and the inputs of the first multiplexer (56_1; 56_2; 56_3; 56_4) in the r:th decompression logic means (50_1; 50_2; 50_3; 50_4) is connected to the pointer output of the r:th memory means (36_1; 36_2; 36_3; 36_4) and to a logically 0-signal, and the inputs of the second multiplexer (58_1; 58_2; 58_3; 58_4) in the r:th decompression logic means (50_1; 50_2; 50_3; 50_4 is connected to the output from the AND-gate (52_1; 52_2; 52_3; 52_4) in the r:th decompression logic means (50_1; 50_2; 50_3; 50_4), and to an output from the comparator (54_1; 54_2; 54_3; 54_4) in the r:th decompression logic means (50_1; 50_2; 50_3; 50_4), and in that the new pointer output from the r:th decompression logic means (50_1; 50_2; 50_3; 50_4) either is the output from the first multiplexer (56_1; 56_2; 56_3; 56_4) or the output from the second multiplexer (58_1; 58_2; 58_3; 58_4) in the r:th decompression logic means (50_1; 50_2; 50_3; 50_4) depending on if the compress flag bit is set or not.

10. A method for classification and forwarding of packets, wherein a destination address of an incoming packet comprises n bits, wherein n is an integer, **characterized in** that the method comprises i stages, wherein each stage represents a predetermined prefix length and a prefix represents a group of addresses, wherein the first stage represents the shortest prefix length and the i:th stage represents the longest prefix length with n bits, wherein each stage comprises a table

of entries, wherein each entry comprises a pointer field which either comprises a pointer to the table in the next stage, or a pointer to a routing table storing the forwarding information, wherein the method comprises the steps;

- to search for a first number of bits corresponding to the shortest prefix length of the destination address in the first table;
- if the entry in the first table wherein a match is found comprises a pointer to the second table, to search for a second number of bits corresponding to the next shortest prefix length of the destination address in the second table;
- and so on until a longest matching prefix has been found, wherein the pointer to the routing table gives the forwarding information for forwarding of said incoming packet.

11. A method according to Claim 10, **characterized in** that each entry in the first (i-1) tables also comprises a part bit and a valid bit, and each entry in the i:th table also comprises a valid bit, wherein a set valid bit in an entry represents a valid prefix, and the pointer field for that entry comprises a pointer to the routing table, and a set part in an entry in the q:th table means that the prefix matches an entry in the routing table, but is shorter than that entry, and the pointer field for the entry in the q:th table comprises a pointer to the (q+1):th table, wherein q is an integer and $1 \leq q \leq i-1$.

12. A method according to Claim 11, **characterized in** that an entry in the first (i-1) tables is not allowed to both have a set valid bit and a set part bit.

13. A method according to Claim 12, **characterized in** that a match is found when an entry in one stage comprises a set valid bit and all the matching entries of shorter prefix lengths are set part bits.

14. A method according to Claim 13, **characterized in** that each entry in each table also comprises a mask field, an address tag field, and a compress flag bit, wherein a set compress flag bit indicates that compression is used, wherein the method for the r:th table, wherein r is an integer and $1 \leq r \leq i$, also comprises the following steps:

- to AND-process the mask field and the destination address bits corresponding to the r:th prefix length, giving masked destination address bits as output;
- to compare the masked destination address bits to the address tag field;
- if the compress flag bit is not set, to output a new pointer which is input to the q:th table, and which is input to the routing table for the i:th table, wherein the new pointer is formed by using the pointer from the (q-1):th table unchanged, with the destination address bits corresponding to the r:th prefix length as low order bits; or
- if the compress flag bit is set, to output a new pointer which is input to the q:th table, and which is input to the routing table for the i:th table, wherein the new pointer is formed with the high order bits set to 0, and the low order bits are formed using the bits of the high order part of the pointer from the (q-1):th table, and the least significant bit of the new pointer comes from the comparing step.

Fig. 1

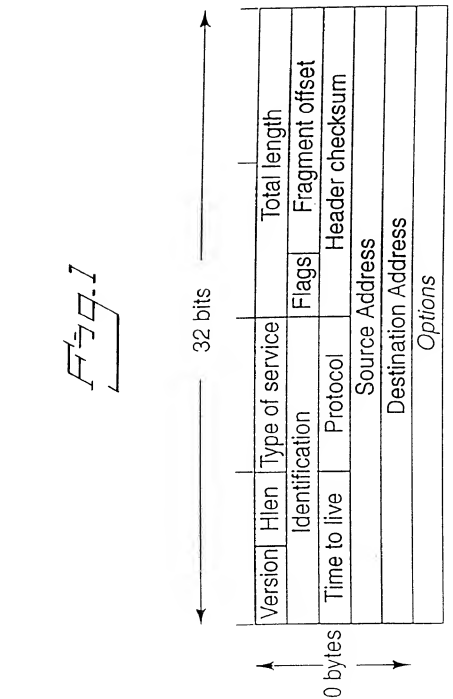
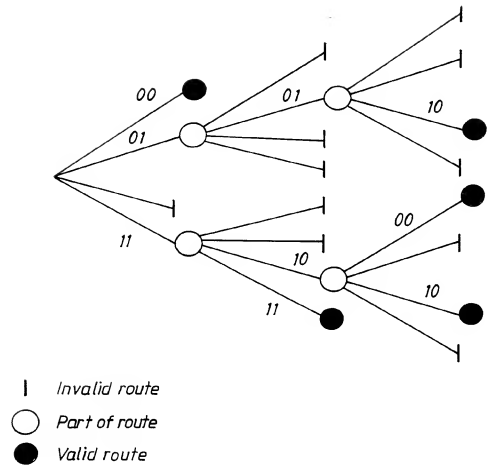

32 bits
20 bytes
Version
Hlen
Type of service
Total length
Identification
Flags
Fragment offset
Time to live
Protocol
Header checksum
Source Address
Destination Address
Options

Fig. 2


00
01
11
01
10
11
00
10
10
Invalid route
Part of route
Valid route

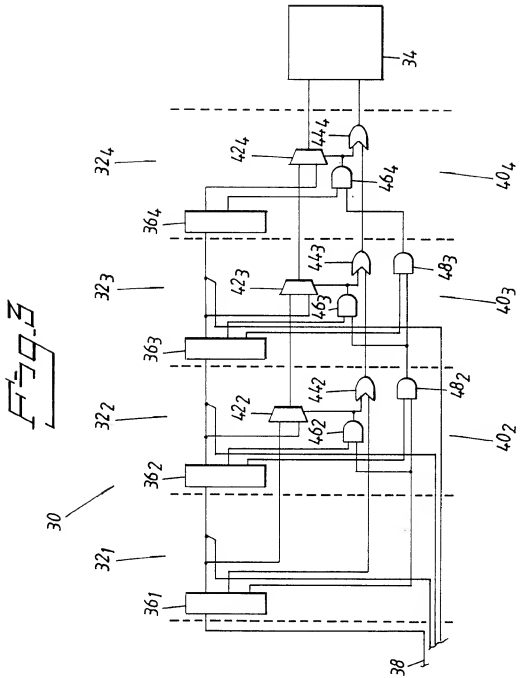

34
324
424
444
464
404
364
323
423
443
463
483
403
363
322
422
442
462
482
402
362
30
321
361
38

FIG.3

Fig. 4

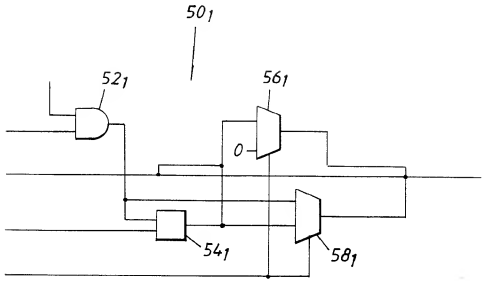
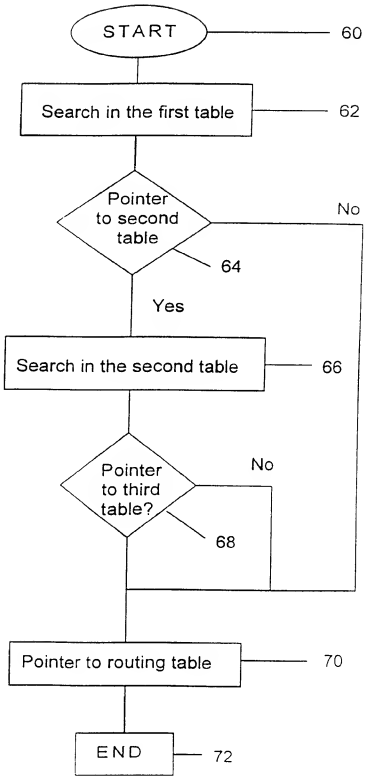


Fig.5


START
60
Search in the first table
62
Pointer to second table
64
No
Yes
Search in the second table
66
Pointer to third table?
68
No
Pointer to routing table
70
END
72